



```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII
SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSSS
SSSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```



```
0001 0 %TITLE 'FORMAT - generate formatted output lines'
0002 0 !<BLF/NOFORMAT>
0003 0
0004 0 MODULE format (IDENT = 'V04-000'
0005 0 %BLISS32[, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE)]
0006 0 ) =
0007 1 BEGIN
0008 1
0009 1 <BLF/FORMAT>
0010 1 <BLF/LOWERCASE_USER>
0011 1 <BLF/UPPERCASE_KEY>
0012 1 <BLF/MACRO>
0013 1
0014 1
0015 1 *****
0016 1 *
0017 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0018 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0019 1 * ALL RIGHTS RESERVED.
0020 1 *
0021 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0022 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0023 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0024 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0025 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0026 1 * TRANSFERRED.
0027 1 *
0028 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0029 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0030 1 * CORPORATION.
0031 1 *
0032 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0033 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0034 1 *
0035 1 *
0036 1 *****
0037 1
0038 1
0039 1 ++
0040 1 FACILITY:
0041 1 DSR (Digital Standard RUNOFF) /DSRPLUS DSRTOC/CONTENTS Utility
0042 1
0043 1 ABSTRACT:
0044 1 Generate formatted output lines
0045 1
0046 1 ENVIRONMENT: Transportable
0047 1
0048 1 AUTHOR: JPK
0049 1
0050 1 CREATION DATE: March 1982
0051 1
0052 1 MODIFIED BY:
0053 1
0054 1 005 JPK00008 09-Mar-1983
0055 1 Modified CONTENTS and CAPTION to support new BRN formats,
0056 1 support SEND CONTENTS, /DOUBLE_SPACE, page numbered chapters,
0057 1 guarantee space after section number and to write new
```



```

58 0058 1 | prologue and epilog for RUNOFF output.
59 0059 1 | Modified FORMAT to quote only the RUNOFF flags used by CONTENTS.
60 0060 1 | Modified CNTVMS to fix default for /DOUBLE_SPACE and do more
61 0061 1 | value checking.
62 0062 1 |
63 0063 1 | 004 JPK00007 14-Feb-1983
64 0064 1 | Global edit of all sources for CONTENTS/DSRTOC:
65 0065 1 | - module names are now consistant with file names
66 0066 1 | - copyright dates have been updated
67 0067 1 | - facility names have been updated
68 0068 1 | - revision history was updated to be consistant with DSR/DSRPLUS
69 0069 1 |
70 0070 1 | 003 JPK00006 14-Feb-1983
71 0071 1 | Modified CNTVMS, CONTENTS, FORMAT and CNTVMSMSG to generate
72 0072 1 | error messages for DSRTOC or CONTENTS depending on the
73 0073 1 | compiletime variant for DSRPLUS (/VARIANT:8192)
74 0074 1 |
75 0075 1 | 002 JPK00004 11-Feb-1983
76 0076 1 | Changed the global variable name INDENT to LINE_INDENT in
77 0077 1 | modules CONTENTS, CAPTION, FORMAT and GBLDCL.
78 0078 1 | Removed declarations of PDENTS in modules CNTVMS, CONTENTS,
79 0079 1 | and CAPTION and replaced with a module wide BIND using the
80 0080 1 | new name INDENTS.
81 0081 1 | Changed handling of INDENTS [1]. It no longer represents the
82 0082 1 | sum of the chapter and title indents.
83 0083 1 |
84 0084 1 | --
85 0085 1 |
86 0086 1 |
87 0087 1 | TABLE OF CONTENTS:
88 0088 1 |
89 0089 1 |
90 0090 1 | FORWARD ROUTINE
91 0091 1 | insref : NOVALUE, ! Insert a page reference
92 0092 1 | fmttxt : NOVALUE, ! Format and output text
93 0093 1 | endwrđ : NOVALUE, ! Verify word fits on line
94 0094 1 | split : NOVALUE; ! Start new output file for TMS
95 0095 1 |
96 0096 1 |
97 0097 1 | INCLUDE FILES:
98 0098 1 |
99 0099 1 |
100 0100 1 | LIBRARY 'NXPORT:XPORT';
101 0101 1 |
102 L 0102 1 | %IF %BLISS (BLISS32)
103 0103 1 | %THEN
104 0104 1 |
105 0105 1 | REQUIRE 'REQ:CNVMSREQ';
106 0345 1 |
107 0346 1 | %FI
108 0347 1 |
109 0348 1 | REQUIRE 'REQ:TOCRTY'; ! Table of Contents file formats
110 0458 1 |
111 0459 1 | REQUIRE 'REQ:CNTCLI'; ! Command line information block formats
112 0587 1 |
113 0588 1 |
114 0589 1 |

```



```

115      0590 1  | MACROS:
116      0591 1  |
117      0592 1  |
118      0593 1  | MACRO
119      0594 1  |
120      0595 1  |     Write a character to output line
121      0596 1  |
122      M 0597 1  |     write_char (ch) [] =
123      M 0598 1  |         BEGIN
124      M 0599 1  |             CH$WCHAR_A (ch, lp);
125      M 0600 1  |             intlin = .intlin + 1;
126      M 0601 1  |
127      M 0602 1  |             %IF NOT %NULL (%REMAINING)
128      M 0603 1  |             %THEN
129      M 0604 1  |                 extlin = .extlin + 1;
130      M 0605 1  |             %FI
131      M 0606 1  |
132      M 0607 1  |         END
133      M 0608 1  |     %,
134      M 0609 1  |     Write a text literal to the output line
135      M 0610 1  |
136      M 0611 1  |     literal_text (str) =
137      M 0612 1  |         BEGIN
138      M 0613 1  |             CH$MOVE (%CHARCOUNT (str), CH$PTR (UPLIT (str)), .lp);
139      M 0614 1  |             lp = CH$PLUS (.lp, %CHARCOUNT (str));
140      M 0615 1  |             intlin = .intlin + %CHARCOUNT (str);
141      M 0616 1  |
142      M 0617 1  |         END
143      M 0618 1  |     %,
144      M 0619 1  |     Pad the output line with blanks
145      M 0620 1  |
146      M 0621 1  |     pad (n_blanks) =
147      M 0622 1  |         BEGIN
148      M 0623 1  |
149      M 0624 1  |             IF n_blanks GTR 0
150      M 0625 1  |             THEN
151      M 0626 1  |                 BEGIN
152      M 0627 1  |                     CH$FILL (%C' ', n_blanks, .lp);
153      M 0628 1  |                     lp = CH$PLUS (.lp, n_blanks);
154      M 0629 1  |                     intlin = .intlin + n_blanks;
155      M 0630 1  |                     extlin = .extlin + n_blanks;
156      M 0631 1  |                 END;
157      M 0632 1  |
158      M 0633 1  |             END
159      M 0634 1  |     %,
160      M 0635 1  |     Clear the text lines being built up.
161      M 0636 1  |
162      M 0637 1  |     clr_line (_) =
163      M 0638 1  |         BEGIN
164      M 0639 1  |             lp = CH$PTR (line);
165      M 0640 1  |             intlin = 0;
166      M 0641 1  |             extlin = 0;
167      M 0642 1  |
168      M 0643 1  |         END
169      M 0644 1  |     %,
170      M 0645 1  |     !
171      0646 1  |

```



```

172      0647 1      ! Insert specified character sequence into file, as is.
173      0648 1
174      M 0649 1      put (str) =
175      M 0650 1          BEGIN
176      M 0651 1          $str_copy (string = str, target = tmpstr);
177      M 0652 1          chROUT = .chROUT + .tmpstr [str$length];
178      M 0653 1          $xpo_put (iob = tocoob, string = tmpstr);
179      M 0654 1
180      M 0655 1          ! For TMS output, split the output file if it gets too large
181      M 0656 1
182      M 0657 1
183      M 0658 1          IF .cmdblk [contents$v_tms11] THEN split ();
184      M 0659 1
185      M 0660 1      END
186      0661 1      %;
187      0662 1
188      0663 1      !
189      0664 1      ! EQUATED SYMBOLS:
190      0665 1      !
191      0666 1
192      0667 1      LITERAL
193      0668 1          tms_characters_per_file = 20*512,          ! TMS files may be 20 blocks long
194      0669 1          rintes = %0'34' : UNSIGNED (8),
195      0670 1          true = 1,
196      0671 1          false = 0;
197      0672 1
198      0673 1      !
199      0674 1      ! OWN STORAGE:
200      0675 1      !
201      0676 1
202      0677 1      OWN
203      0678 1          fileno : INITIAL (0),
204      0679 1          outfile : $str_descriptor (class = dynamic, string = (0,0)), ! Save output filename here
205      0680 1          wrdptr,
206      0681 1          extwr,
207      0682 1          intwr;
208      0683 1
209      0684 1      !
210      0685 1      ! EXTERNAL REFERENCES:
211      0686 1      !
212      0687 1
213      0688 1      EXTERNAL
214      0689 1          cmdblk : $contents_cmd,
215      0690 1          tocoob : $xpo_iob (?),
216      0691 1          chROUT,
217      0692 1          tmpstr : $str_descriptor (),
218      0693 1          hl_n,
219      0694 1          major,
220      0695 1          lp,
221      0696 1          intlin,
222      0697 1          extlin,
223      0698 1          line : VECTOR [CH$ALLOCATION (10000)],
224      0699 1          lenpag,
225      0700 1          txtpag : VECTOR [CH$ALLOCATION (50)],
226      0701 1          rmargin,
227      0702 1          wrap,
228      0703 1          line_indent;

! Storage for remembering words.
! Output file number
! CH$PTR to start of current word.
! Number of print positions in current word.
! Number of characters needed to represent current word.

! Command line information block
! IOB for the resulting .RNT file
! Number of characters written to output file
! For temporary strings
! "n" from latest .HL n command
! Major record type code
! CH$PTR along line being built up
! Number of characters needed to represent text
! Number of resulting print positions
! Buffer in which line is being built up.
! Number of characters in the converted page number.
! The text (lots of room)
! Used by ENDWRD for controlling filling lines.
! Wrap long lines around to here.
! Assume this standard indentation before the text.

```



FORMAT  
V04-000

FORMAT - generate formatted output lines

F 13  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 5  
(1)

```
: 229
: 230
: 231
: 232
: 233
: 234
: 235
: 236

      0704 1
L 0705 1 %IF %BLISS (BLISS32)
      0706 1 %THEN
      0707 1
      0708 1 EXTERNAL ROUTINE
      0709 1      open_error;
      0710 1
      0711 1 %FI
```

! File open error handler

FOR  
V04

```

238 0712 1 %SBTTL 'INSREF - insert page reference into line'
239 0713 1 GLOBAL ROUTINE insref : NOVALUE =
240 0714 1 ++
241 0715 1
242 0716 1 FUNCTIONAL DESCRIPTION:
243 0717 1
244 0718 1 This routine inserts a page reference into the output line
245 0719 1
246 0720 1 FORMAL PARAMETERS:
247 0721 1
248 0722 1 None
249 0723 1
250 0724 1 IMPLICIT INPUTS:
251 0725 1
252 0726 1 cmdblk - command line information block
253 0727 1 hl_n - current header level
254 0728 1 rmargin - right margin
255 0729 1 extlin - external line length
256 0730 1
257 0731 1 IMPLICIT OUTPUTS:
258 0732 1
259 0733 1 wrap - line wrap point
260 0734 1 rmargin - right margin
261 0735 1 - variables related to output line
262 0736 1
263 0737 1 ROUTINE VALUE:
264 0738 1 COMPLETION CODES:
265 0739 1
266 0740 1 None
267 0741 1
268 0742 1 SIDE EFFECTS:
269 0743 1
270 0744 1 None
271 0745 1 --
272 0746 1
273 0747 2 BEGIN
274 0748 2
275 0749 2 IF .hl_n GTR .cmdblk [contents$g_page_level] THEN RETURN;
276 0750 2
277 0751 2 IF .cmdblk [contents$g_tms11]
278 0752 2 THEN
279 0753 3 write_char (%C'a')
280 0754 3
281 0755 3 ELSE
282 0756 3 BEGIN
283 0757 3
284 0758 3 OK. User wants this header to show dots and page number.
285 0759 3 Insert a sequence of alternating dots and spaces out to where the
286 0760 3 page number will go.
287 0761 3 First force a space to follow the last text character.
288 0762 3
289 0763 3 IF .extlin LSS .rmargin
290 0764 3 THEN
291 0765 4 BEGIN
292 0766 4
293 0767 4 The position of the last character of the text
294 0768 4 is not inside where the page number goes.

```



```

295 0769 4
296 0770 4
297 0771 4
298 0772 4
299 0773 4
300 0774 4
301 0775 4
302 0776 4
303 0777 4
304 0778 4
305 0779 4
306 0780 4
307 P 0781 4
308 0782 4
309 0783 4
310 0784 4
311 0785 4
312 0786 4
313 0787 4
314 0788 4
315 0789 4
316 0790 4
317 0791 4
318 0792 4
319 0793 4
320 0794 4
321 0795 4
322 0796 4
323 0797 4
324 0798 4
325 0799 4
326 0800 2
327 0801 2
328 0802 2
329 0803 2
330 0804 2
331 0805 2
332 0806 2
333 0807 2
334 0808 1

```

```

!
write_char (%C' ', counts_visually);
END;

!
Now start inserting the dot-space sequence
!
INCR i FROM (.extlin + 1) TO .rmargin DO
!
Insert a space for odd columns, a dot for even ones.
!
write_char ((IF .i THEN %C' ' ELSE CH$RCHAR (CH$PTR (cmdblk [contents$c_leader_char]))),
counts_visually);
!
Insert a space following the sequence so there's
no dot just before the page number.
!
write_char (%C' ', counts_visually);
!
Before putting the page number through the standard text processor,
set the wrap point in case the page number doesn't fit.
!
wrap = .cmdblk [contents$g_page_width] - .lenpag;
!
Allow the page number to cause the line to be
filled out to the full page width.
!
rmargin = .cmdblk [contents$g_page_width];
END;

!
Finally, insert the generated page number onto the line
!
fmttxt (.lenpag, CH$PTR (txtpag));
!
IF .cmdblk [contents$v_tms11] THEN write_char (%C'a');
END;

```

```

.TITLE FORMAT FORMAT - generate formatted output lines
.IDENT \V04-000\
.PSECT $OWNS,NOEXE,2
00000000 00000 FILENO: .LONG 0
0000 00004 OUTFILE: .WORD 0
02 0E 00006 .BYTE 14, 2
00000000 00008 .LONG 0
0000C WRDPTR: .BLKB 4
00010 EXTWRD: .BLKB 4
00014 INTWRD: .BLKB 4
.EXTRN DSRTOC$_BADVALUE
.EXTRN DSRTOC$_OPENIN, DSRTOC$_OPENOUT

```





FORMAT  
V04-000

FORMAT - generate formatted output lines  
INSREF - insert page reference into line

J 13  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 9  
(2)

	66		18	A3	D0	0007E		MOVL	CMDBLK+24, RMARGIN		: 0798
				62	D6	00082	7\$:	INCL	LP		: 0753
				65	D6	00084		INCL	INTLIN		: 0804
			00000000G	EF	9F	00086		PUSHAB	TXTPAG		: 0806
				67	DD	0008C		PUSHL	LENPAG		: 0808
09	00000000V	EF		02	FB	0008E		CALLS	#2, FMTTXT		: 0806
		63		01	E1	00095		BBC	#1, CMDBLK, 8\$		: 0808
	00	B2		8F	90	00099		MOVB	#64, @LP		: 0808
			40	62	D6	0009E		INCL	LP		: 0808
				65	D6	000A0		INCL	INTLIN		: 0808
					04	000A2	8\$:	RET			: 0808

; Routine Size: 163 bytes, Routine Base: \$CODE\$ + 0000

```

336 0809 1 %SBTTL 'FMTTXT - scan and format text'
337 0810 1 GLOBAL ROUTINE fmttxt (txt_len, txt_ptr) : NOVALUE =
338 0811 1 ++
339 0812 1
340 0813 1 FUNCTIONAL DESCRIPTION:
341 0814 1
342 0815 1     This routine scans the input text and formats it into line.
343 0816 1
344 0817 1     Special characters are quoted for RUNOFF output or changed to the
345 0818 1     appropriate sequence for TMS unless the text is from a .SEND TOC.
346 0819 1
347 0820 1     Special characters from .SEND TOC are inserted as is for RUNOFF.
348 0821 1
349 0822 1     Special characters from .SEND TOC are inserted as is for TMS
350 0823 1     with the exception of '>' which is inserted as '>', the line
351 0824 1     is broken, and a new line is started with '<'. This is because
352 0825 1     .SEND TOC text is inserted as a comment for TMS.
353 0826 1
354 0827 1     Emphasis in the input string is kept if emphasis is enabled
355 0828 1     for the current header level or if the text is from a .SEND TOC.
356 0829 1
357 0830 1 FORMAL PARAMETERS:
358 0831 1
359 0832 1     txt_len      - length of input string
360 0833 1     txt_ptr      - CH$PTR to input string
361 0834 1
362 0835 1 IMPLICIT INPUTS:
363 0836 1
364 0837 1     cmdblk      - command line information block
365 0838 1     major      - type of text being processed
366 0839 1     hl_n       - header level number being processed
367 0840 1     lp        - CH$PTR to next character position in line
368 0841 1     intlin     - internal line length
369 0842 1     extlin     - external line length
370 0843 1     rmargin    - indicates how far to the right text may be inserted
371 0844 1     wrap      - column to wrap a broken line to
372 0845 1     line_indent - number of columns which line is indented
373 0846 1
374 0847 1 IMPLICIT OUTPUTS:
375 0848 1
376 0849 1     lp        - points to next available character position in line
377 0850 1     intlin     - reflects new internal length
378 0851 1     extlin     - reflects new external length
379 0852 1     wrdptr     - set to initial value of lp
380 0853 1     intwrld    - set to initial value of intlin
381 0854 1     extwrld    - set to initial value of extlin
382 0855 1
383 0856 1 ROUTINE VALUE:
384 0857 1 COMPLETION CODES:
385 0858 1
386 0859 1     None
387 0860 1
388 0861 1 SIDE EFFECTS:
389 0862 1
390 0863 1     None
391 0864 1 --
392 0865 1

```



```

: 393      0866      2      BEGIN
: 394      0867      2
: 395      0868      2      LOCAL
: 396      0869      2          keep_bold,
: 397      0870      2          keep_und,
: 398      0871      2          doing_bold,
: 399      0872      2          doing_und,
: 400      0873      2          bold_char,
: 401      0874      2          und_char,
: 402      0875      2          open_quote,
: 403      0876      2          ptr,
: 404      0877      2          len;
: 405      0878      2
: 406      0879      2      !
: 407      0880      2      ! Keep bolding if .SEND TOC and user said /BOLD=anything
: 408      0881      2      ! or if the hl value of the text LEQ the user specified level.
: 409      0882      2      !
: 410      0883      2      IF ((.major EQL maj_send) AND (.cndblk [contents$g_bold] NEQ -1))
: 411      0884      2          OR (.hl_n LEQ .cndblk [contents$g_bold])
: 412      0885      2      THEN
: 413      0886      2          keep_bold = true
: 414      0887      2      ELSE
: 415      0888      2          keep_bold = false;
: 416      0889      2
: 417      0890      2      !
: 418      0891      2      ! Keep underlining if .SEND TOC and user said /UNDERLINE=anything
: 419      0892      2      ! or if the hl value of the text LEQ the user specified level.
: 420      0893      2      !
: 421      0894      2      IF ((.major EQL maj_send) AND (.cndblk [contents$g_underline] NEQ -1))
: 422      0895      2          OR (.hl_n LEQ .cndblk [contents$g_underline])
: 423      0896      2      THEN
: 424      0897      2          keep_und = true
: 425      0898      2      ELSE
: 426      0899      2          keep_und = false;
: 427      0900      2
: 428      0901      2      len = .txt_len;
: 429      0902      2      ptr = .txt_ptr;
: 430      0903      2      wrdptr = .tp;
: 431      0904      2      intwrd = .intlin;
: 432      0905      2      extwrd = .extlin;
: 433      0906      2      doing_bold = false;
: 434      0907      2      doing_und = false;
: 435      0908      2      bold_char = false;
: 436      0909      2      und_char = false;
: 437      0910      2      open_quote = true;
: 438      0911      2
: 439      0912      2      WHILE .len GTR 0 DO
: 440      0913      2          BEGIN
: 441      0914      2
: 442      0915      2          LOCAL
: 443      0916      2              ch;
: 444      0917      2
: 445      0918      2          ch = CH$RCHAR A (ptr);
: 446      0919      2          len = .len - 1;
: 447      0920      2
: 448      0921      2          IF .ch EQL rintex
: 449      0922      2          THEN

```

```

! Copy string length
! and pointer
! Initialize word pointer
! internal word length
! external word length
! Bold is off
! as is underlining
! Character is not bold
! or underlined
! First "" we see is an open quote
! Process whole input string
! Get next character
! one less character

```

```

: 450      0923 4      BEGIN
: 451      0924 4
: 452      0925 4      | RUNOFF internal escape sequence
: 453      0926 4
: 454      0927 4
: 455      0928 4      LOCAL
: 456      0929 4      fnc,
: 457      0930 4      op;
: 458      0931 4
: 459      0932 4      fnc = CH$RCHAR_A (ptr);      | Get function
: 460      0933 4      op = CH$RCHAR_A (ptr);      | and operand
: 461      0934 4      len = .len - 2;      | 2 less characters to process
: 462      0935 4
: 463      0936 4      SELECTONE .fnc OF
: 464      0937 4      SET
: 465      0938 4
: 466      0939 4      [%C'O'] :
: 467      0940 5      BEGIN
: 468      0941 5      | Overstrike
: 469      0942 5      |
: 470      0943 5      | write_char (.op, counts_visually);
: 471      0944 5
: 472      0945 5      IF .cmdblk [contents$v_tms11]
: 473      0946 5      THEN
: 474      0947 5      BEGIN
: 475      0948 6      | Overstriking is frowned upon for TMS
: 476      0949 6      |
: 477      0950 6
: 478      0951 6
: 479      L 0952 6 %IF %BLISS (BLISS32)
: 480      0953 6 %THEN
: 481      0954 6      SIGNAL (contents$_overstrk, 0, contents$_textd, 2, .txt_len, .txt_ptr);
: 482      U 0955 6 %ELSE
: 483      0956 6      $xpo_put_msg (severity = warning,
: 484      0957 6      string = 'the following line contains an overstrike sequence',
: 485      0958 6      string = (.txt_len, .txt_ptr));
: 486      0959 6 %FI
: 487      0960 6
: 488      0961 6      literal_text ('[ec]');
: 489      0962 6      END
: 490      0963 5      ELSE
: 491      0964 5      write_char (%C'%');
: 492      0965 5
: 493      0966 4      END;
: 494      0967 4
: 495      0968 4      [%C'B'] :
: 496      0969 4      | Bold next character if keeping bold
: 497      0970 4      |
: 498      0971 4      | bold_char = (IF .keep_bold THEN true ELSE false);
: 499      0972 4
: 500      0973 4
: 501      0974 4      [%C'U'] :
: 502      0975 4      | Underline next character if keeping underlining
: 503      0976 4      |
: 504      0977 4      | und_char = (IF .keep_und THEN true ELSE false);
: 505      0978 4
: 506      0979 4

```



```

507      0980 4      [OTHERWISE] :
508      0981 4      |
509      0982 4      | Unknown sequence - do nothing
510      0983 4      |
511      0984 4      |
512      0985 4      | YES;
513      0986 4      |
514      0987 4      |
515      0988 3      |
516      0989 4      |
517      0990 4      |
518      0991 4      | A 'normal' character
519      0992 4      |
520      0993 4      |
521      0994 4      | IF NOT .bold_char
522      0995 4      | THEN
523      0996 5      | BEGIN
524      0997 5      |
525      0998 5      | Do not bold this character
526      0999 5      |
527      1000 5      |
528      1001 6      | IF .doing_bold AND (.ch NEQ %C' ')
529      1002 5      | THEN
530      1003 6      | BEGIN
531      1004 6      |
532      1005 6      | Bold is turned on and the current character is non-blank
533      1006 6      | Turn off bold
534      1007 6      |
535      1008 6      |
536      1009 6      | IF .cmdblk [contents$V_tms11] THEN literal_text ('[fr') ELSE literal_text ('\*');
537      1010 6      |
538      1011 6      | IF .doing_und
539      1012 6      | THEN
540      1013 7      | BEGIN
541      1014 7      |
542      1015 7      | Must turn underlining off too on since both bold
543      1016 7      | and underline use the same termination sequence
544      1017 7      |
545      1018 7      |
546      1019 7      | IF .cmdblk [contents$V_tms11] THEN literal_text ('fr') ELSE literal_text ('\&');
547      1020 7      |
548      1021 7      | IF .und_char
549      1022 7      | THEN
550      1023 8      | BEGIN
551      1024 8      |
552      1025 8      | This character is underlined
553      1026 8      | Turn underlining back on.
554      1027 8      |
555      1028 8      |
556      1029 8      | IF .cmdblk [contents$V_tms11] THEN literal_text ('fi') ELSE literal_text ('\&');
557      1030 8      |
558      1031 8      | END
559      1032 7      | ELSE
560      1033 7      |
561      1034 7      | Character is not underlined
562      1035 7      | Note that we've turned off underlining
563      1036 7      |

```

```

: 564      1037 7      doing_und = false;
: 565      1038 7
: 566      1039 6      END;
: 567      1040 6
: 568      1041 6      IF .cmdblk [contents$V_tms11] THEN write_char (%C'');
: 569      1042 6
: 570      1043 6      doing_bold = false;
: 571      1044 5      END;
: 572      1045 5
: 573      1046 5      END
: 574      1047 4      ELSE
: 575      1048 5      BEGIN
: 576      1049 5      |
: 577      1050 5      | Bold next character
: 578      1051 5      |
: 579      1052 5
: 580      1053 5      IF NOT .doing_bold
: 581      1054 5      THEN
: 582      1055 6      BEGIN
: 583      1056 6      |
: 584      1057 6      | Turn on bolding
: 585      1058 6      |
: 586      1059 6
: 587      1060 6      IF .cmdblk [contents$V_tms11] THEN literal_text ('[fb]') ELSE literal_text ('^*');
: 588      1061 6
: 589      1062 6      doing_bold = true;
: 590      1063 5      END;
: 591      1064 5
: 592      1065 5      bold_char = false;      ! Reset bold character flag
: 593      1066 4      END;
: 594      1067 4
: 595      1068 4      IF NOT .und_char
: 596      1069 4      THEN
: 597      1070 5      BEGIN
: 598      1071 5      |
: 599      1072 5      | Do not underline this character
: 600      1073 5      |
: 601      1074 5
: 602      1075 6      IF .doing_und AND (.ch NEQ %C' ')
: 603      1076 5      THEN
: 604      1077 6      BEGIN
: 605      1078 6      |
: 606      1079 6      | Underlining is turned on and the current character is
: 607      1080 6      | non-blank. Turn off underlining.
: 608      1081 6      |
: 609      1082 6
: 610      1083 6      IF .cmdblk [contents$V_tms11] THEN literal_text ('[fr'] ELSE literal_text ('\&');
: 611      1084 6
: 612      1085 6      IF .cmdblk [contents$V_tms11]
: 613      1086 6      THEN
: 614      1087 7      BEGIN
: 615      1088 7      |
: 616      1089 7      | If bolding is on, turn it off and back on since
: 617      1090 7      | both bold and underline use the same terminators
: 618      1091 7      |
: 619      1092 7
: 620      1093 7      IF .doing_bold THEN literal_text ('frfb');

```



```

621 1094 7
622 1095 7
623 1096 6
624 1097 6
625 1098 6
626 1099 5
627 1100 5
628 1101 5
629 1102 4
630 1103 5
631 1104 5
632 1105 5
633 1106 5
634 1107 5
635 1108 5
636 1109 5
637 1110 6
638 1111 6
639 1112 6
640 1113 6
641 1114 6
642 1115 6
643 1116 6
644 1117 6
645 1118 5
646 1119 5
647 1120 5
648 1121 4
649 1122 4
650 1123 4
651 1124 4
652 1125 4
653 1126 4
654 1127 4
655 1128 4
656 1129 4
657 1130 5
658 1131 5
659 1132 5
660 1133 5
661 1134 6
662 1135 6
663 1136 6
664 1137 6
665 1138 6
666 1139 7
667 1140 7
668 1141 7
669 1142 7
670 1143 7
671 1144 7
672 1145 7
673 1146 7
674 1147 7
675 1148 7
676 1149 6
677 1150 6

write_char (%C']');
END;

doing_und = false;
END;

END
ELSE
BEGIN
    Underline next character

    IF NOT .doing_und
    THEN
        BEGIN
            Turn on underlining

            IF .cmdblk [contents$V_tms11] THEN literal_text ('[fi]') ELSE literal_text ('^&');

            doing_und = true;
            END;

            und_char = false;                ! Reset flag
            END;

        SELECTONE true OF
        SET
            [.ch EQL %C' '] :
                endwrd (true);

            [.ch EQL %C'>'] :
                BEGIN
                    IF .major EQL maj_send
                    THEN
                        BEGIN
                            write_char (.ch, counts_visually);

                            IF .cmdblk [contents$V_tms11]
                            THEN
                                BEGIN
                                    For TMS, a '>' in a SEND TOC starts a new line

                                    put ((.intlín, CH$PTR (line)));
                                    clr_line ();
                                    write_char (%C'<');
                                    wrdptr = .lp;
                                    extwrd = .extlin;
                                    intwrd = .intlín;
                                    END;
                                END;
                            END;
                        END;
                    END;
                END;
            END;
        END;
    END;
END;

```

FORMAT  
V04-000

FORMAT - generate formatted output lines  
FMTTXT - scan and format text

D 14  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 16  
(3)

```

: 678      1151 6      END
: 679      1152 5      ELSE
: 680      1153 6      BEGIN
: 681      1154 6
: 682      1155 6      IF .cndblk [contents$V_tms11]
: 683      1156 6      THEN
: 684      1157 7      literal_text ('+z')
: 685      1158 6      ELSE
: 686      1159 6      write_char (.ch, counts_visually);
: 687      1160 6
: 688      1161 5      END;
: 689      1162 5
: 690      1163 4      END;
: 691      1164 4
: 692      1165 4      [(.ch LSS %C' ') OR (.ch GTR %O'176')] :
: 693      1166 5      BEGIN
: 694      1167 5      |
: 695      1168 5      | A control character
: 696      1169 5      |
: 697      1170 5
: 698      1171 5      IF .cndblk [contents$V_tms11]
: 699      1172 5      THEN
: 700      1173 6      BEGIN
: 701      1174 6      |
: 702      1175 6      | Control characters are ignored for TMS output
: 703      1176 6      |
: 704      L 1177 6 %IF %BLISS (BLISS32)
: 705      1178 6 %THEN
: 706      1179 6      SIGNAL (contents$_ctrlchar, 0, contents$_textd, 2, .txt_len, .txt_ptr);
: 707      U 1180 6 %ELSE
: 708      U 1181 6
: 709      U 1182 6      $xpo_put_msg (severity = warning,
: 710      U 1183 6      string = 'the following line contains control characters which were ignored',
: 711      1184 6      string = (.txt_len, .txt_ptr));
: 712      1185 6 %FI
: 713      1186 6
: 714      1187 5      ELSE
: 715      1188 6      BEGIN
: 716      1189 6      |
: 717      1190 6      | For RUNOFF output
: 718      1191 6      |
: 719      1192 6      write_char (%C' ');      ! Quote the character
: 720      1193 6      write_char (.ch);      ! Write the character itself
: 721      1194 6
: 722      1195 6      IF .ch EQL %O'10'
: 723      1196 6      THEN
: 724      1197 6      extlin = .extlin - 1;      ! Backspace shortens the external length
: 725      1198 6
: 726      1199 5      END;
: 727      1200 5
: 728      1201 4      END;
: 729      1202 4
: 730      1203 4      [OTHERWISE] :
: 731      1204 5      BEGIN
: 732      1205 5      |
: 733      1206 5      | For every thing else...
: 734      1207 5      |

```



```

: 735 1208 5
: 736 1209 5
: 737 1210 5
: 738 1211 6
: 739 1212 6
: 740 1213 6
: 741 1214 6
: 742 1215 6
: 743 1216 6
: 744 1217 5
: 745 1218 6
: 746 1219 6
: 747 1220 6
: 748 1221 6
: 749 1222 6
: 750 1223 6
: 751 1224 6
: 752 1225 7
: 753 1226 7
: 754 1227 7
: 755 1228 7
: 756 1229 7
: 757 1230 7
: 758 1231 7
: 759 1232 7
: 760 1233 7
: 761 1234 7
: 762 1235 7
: 763 1236 7
: 764 1237 7
: 765 1238 7
: 766 1239 7
: 767 1240 7
: 768 1241 7
: 769 1242 7
: 770 1243 7
: 771 1244 7
: 772 1245 7
: 773 1246 7
: 774 1247 7
: 775 1248 7
: 776 1249 7
: 777 1250 7
: 778 1251 7
: 779 1252 7
: 780 1253 7
: 781 1254 7
: 782 1255 7
: 783 1256 7
: 784 1257 7
: 785 1258 7
: 786 1259 7
: 787 1260 7
: 788 1261 7
: 789 1262 7
: 790 1263 7
: 791 1264 7

IF .major EQL maj_send
THEN
BEGIN
: Just write the character for .SEND TOC
: write_char (.ch, counts_visually);
END
ELSE
BEGIN
: Check for special characters
:
IF .cndblk [contents$v_tms11]
THEN
BEGIN
: For TMS...
:
SELECTONE .ch OF
SET
[XC' ']:
literal_text ('*n10*');
[XC'-']:
literal_text ('+n');
[XC'*']:
literal_text ('+a');
[XC'=']:
literal_text ('+e');
[XC'+']:
literal_text ('+p');
[XC'\']:
literal_text ('+s');
[XC'@']:
literal_text ('+t');
[XC'/']:
literal_text ('+.');
[XC'|']:
literal_text ('+v');
[XC'{']:
literal_text ('+w');
[XC')']:
literal_text ('+x');

```

```

: 792      1265 7
: 793      1266 7
: 794      1267 7
: 795      1268 7
: 796      1269 7
: 797      1270 7
: 798      1271 7
: 799      1272 7
: 800      1273 7
: 801      1274 7
: 802      1275 7
: 803      1276 8
: 804      1277 8
: 805      1278 8
: 806      1279 8
: 807      1280 9
: 808      1281 9
: 809      1282 9
: 810      1283 9
: 811      1284 9
: 812      1285 9
: 813      1286 9
: 814      1287 8
: 815      1288 9
: 816      1289 9
: 817      1290 9
: 818      1291 9
: 819      1292 9
: 820      1293 9
: 821      1294 8
: 822      1295 8
: 823      1296 7
: 824      1297 7
: 825      1298 7
: 826      1299 7
: 827      1300 7
: 828      1301 7
: 829      1302 7
: 830      1303 7
: 831      1304 7
: 832      1305 7
: 833      1306 6
: 834      1307 7
: 835      1308 7
: 836      1309 7
: 837      1310 7
: 838      1311 7
: 839      1312 8
: 840      1313 8
: 841      1314 8
: 842      1315 8
: 843      1316 8
: 844      1317 8
: 845      1318 8
: 846      1319 8
: 847      1320 7
: 848      1321 7

      [%C'<'] :
      literal_text ('+y');

      [%C'['] :
      literal_text ('+( ');

      [%C']'] :
      literal_text ('+')');

      [%C'''] :
      BEGIN
      IF .open_quote
      THEN
      BEGIN
      Opening quote of quoted string
      literal_text ('''');
      open_quote = false;      ! Next quote is not an open quote
      END
      ELSE
      BEGIN
      Closing quote
      literal_text ('''');
      open_quote = true;      ! Next quote is open quote
      END;
      END;

      [OTHERWISE] :
      A real normal character
      write_char (.ch, counts_visually);
      TES;

      END
    ELSE
      BEGIN
      For RUNOFF

      IF (.ch EQL %C' ')
      OR (.ch EQL %C'*')
      OR (.ch EQL %C'!')
      OR (.ch EQL %C'.'')
      OR (.ch EQL %C'\')
      OR (.ch EQL %C'%')
      OR (.ch EQL %C'&')
      OR (.ch EQL %C'^')
      THEN
      !
      ACCEPT flag
      BOLD flag
      COMMENT flag
      CONTROL flag
      LOWERCASE flag
      OVERSTRIKE flag
      UNDERLINE flag
      UPPERCASE flag

```



```

849 1322 7      ! A RUNOFF flag. Quote it.
850 1323 7      !
851 1324 7      write_char (%C'_');
852 1325 7
853 1326 7      write_char (.ch, counts_visually);
854 1327 6      END;
855 1328 6
856 1329 5      END;
857 1330 5
858 1331 4      TES;
859 1332 4      END;
860 1333 4
861 1334 4      END;
862 1335 4
863 1336 4      END;
864 1337 4
865 1338 4      endwrd (false);
866 1339 4      ! Check to see if word fits
867 1340 4      IF .cmdblk [contents$v_tms11]
868 1341 4      THEN
869 1342 4      BEGIN
870 1343 4
871 1344 4      IF .doing_bold OR .doing_und
872 1345 4      THEN
873 1346 4      BEGIN
874 1347 4      !
875 1348 4      ! Turn off one of them.
876 1349 4      !
877 1350 4      ! literal_text ('[fr');
878 1351 4      !
879 1352 4      ! If doing both bold and underline, turn off the other.
880 1353 4      !
881 1354 4      !
882 1355 4      IF .doing_bold AND .doing_und THEN literal_text ('fr');
883 1356 4
884 1357 4      write_char (%C']');
885 1358 4      END;
886 1359 4
887 1360 4      IF NOT .open_quote
888 1361 4      THEN
889 1362 4      !
890 1363 4      ! Missing a close quote
891 1364 4      !
892 1365 4      ! IF %BLISS (BLISS32)
893 1366 4      ! THEN
894 1367 4      !     SIGNAL (contents$_closequot, 0, contents$_textd, 2, .txt_len, .txt_ptr);
895 1368 4      ! %ELSE
896 1369 4      !     $xpo_put_msg (severity = warning,
897 1370 4      !         string = 'the following text is missing a close quote',
898 1371 4      !         string = (.txt_len, .txt_ptr));
899 1372 4      ! %FI
900 1373 4
901 1374 4      END
902 1375 4      ELSE
903 1376 4      BEGIN
904 1377 4      !
905 1378 4      ! For RUNOFF

```

FORMAT  
V04-000

FORMAT - generate formatted output lines  
FMTTXT - scan and format text

H 14  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 20  
(3)

```

: 906      1379 3      !
: 907      1380      !
: 908      1381      IF .doing_bold THEN literal_text ('\*'); ! Turn off bolding
: 909      1382      !
: 910      1383      IF .doing_und THEN literal_text ('\&'); ! Turn off underlining
: 911      1384      !
: 912      1385      END;
: 913      1386      !
: 914      1387 1      END;

```

```

                                .PSECT $PLITS$,NOWRT,NOEXE,2
                                .....
00 00 00 2A 5D 63 65 5B 00000 P.AAA: .ASCII \[ec]\
00 00 00 2A 5B 72 66 5B 00004 P.AAB: .ASCII \[fr\<0>
00 00 00 2A 5C 00 2A 5C 00008 P.AAC: .ASCII <92>\*\<0><0>
00 00 00 2A 66 00 72 66 0000C P.AAD: .ASCII \fr\<0><0>
00 00 00 26 5C 00 26 5C 00010 P.AAE: .ASCII <92>\&\<0><0>
00 00 00 69 66 00 69 66 00014 P.AAF: .ASCII \fi\<0><0>
00 00 00 26 5E 00 26 5E 00018 P.AAG: .ASCII \^&\<0><0>
5D 62 66 5B 0001C P.AAH: .ASCII \[fb]\
00 00 00 2A 5E 00020 P.AAI: .ASCII \^*\<0><0>
00 00 00 72 66 5B 00024 P.AAJ: .ASCII \[fr\<0>
00 00 00 26 5C 00028 P.AAK: .ASCII <92>\&\<0><0>
62 66 72 66 0002C P.AAL: .ASCII \frfb\
5D 69 66 5B 00030 P.AAM: .ASCII \[fi]\
00 00 00 26 5E 00034 P.AAN: .ASCII \^&\<0><0>
00 00 00 7A 2B 00038 P.AAO: .ASCII \+z\<0><0>
00 00 00 2A 2A 0003C P.AAP: .ASCII \*n10*\<0><0><0>
00 00 00 6E 2B 00044 P.AAQ: .ASCII \+n\<0><0>
00 00 00 61 2B 00048 P.AAR: .ASCII \+a\<0><0>
00 00 00 65 2B 0004C P.AAS: .ASCII \+e\<0><0>
00 00 00 70 2B 00050 P.AAT: .ASCII \+p\<0><0>
00 00 00 73 2B 00054 P.AAU: .ASCII \+s\<0><0>
00 00 00 74 2B 00058 P.AAV: .ASCII \+t\<0><0>
00 00 00 2E 2B 0005C P.AAW: .ASCII \+.\<0><0>
00 00 00 76 2B 00060 P.AAX: .ASCII \+v\<0><0>
00 00 00 77 2B 00064 P.AAY: .ASCII \+w\<0><0>
00 00 00 78 2B 00068 P.AAZ: .ASCII \+x\<0><0>
00 00 00 79 2B 0006C P.ABA: .ASCII \+y\<0><0>
00 00 00 28 2B 00070 P.ABB: .ASCII \+(\<0><0>
00 00 00 29 2B 00074 P.ABC: .ASCII \+)\<0><0>
00 00 00 22 22 00078 P.ABD: .ASCII \'\<0><0>
00 00 00 27 27 0007C P.ABE: .ASCII \'\'\<0><0>
00 00 00 72 66 5B 00080 P.ABF: .ASCII \[fr\<0>
00 00 00 72 66 00084 P.ABG: .ASCII \fr\<0><0>
00 00 00 2A 5C 00088 P.ABH: .ASCII <92>\*\<0><0>
00 00 00 26 5C 0008C P.ABI: .ASCII <92>\&\<0><0>
                                .....
                                .EXTRN XST$COPY, STR$FAILURE
                                .EXTRN XPOS$PUT, XPOS$FAILURE
                                .....
                                .PSECT $CODE$,NOWRT,2
                                .....
                                OFFC 00000
                                .ENTRY FMTTXT, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,- ; 0810
                                R11
                                .....

```



5E	18	C2	00002	SUBL2	#24, SP	:	0883
03 00000000G	50	D4	00005	CLRL	RO	:	
	EF	D1	00007	CMPL	MAJOR, #3	:	
	OF	12	0000E	BNEQ	1\$	:	
FFFFFFF	50	D6	00010	INCL	RO	:	
8F 00000000G	EF	D1	00012	CMPL	CMDBLK+16, #-1	:	
	OD	12	0001D	BNEQ	2\$	:	
00000000G	EF	D1	0001F	CMPL	HL_N, CMDBLK+16	:	0884
	05	14	0002A	BGTR	3\$	:	
6E	01	D0	0002C	MOVL	#1, KEEP_BOLD	:	0886
	02	11	0002F	BRB	4\$	:	
	6E	D4	00031	CLRL	KEEP_BOLD	:	0888
OD	50	E9	00033	BLBC	RO, 5\$	:	0894
8F 00000000G	EF	D1	00036	CMPL	CMDBLK+12, #-1	:	
	OD	12	00041	BNEQ	6\$	:	
00000000G	EF	D1	00043	CMPL	HL_N, CMDBLK+12	:	0895
	06	14	0004E	BGTR	7\$	:	
OC AE	01	D0	00050	MOVL	#1, KEEP_UND	:	0897
	03	11	00054	BRB	8\$	:	
	OC	AE	D4	00056	CLRL	KEEP_UND	0899
	5B	04	AC	D0	00059	8\$:	0901
04	AE	08	AC	D0	0005D		0902
00000000'	EF	00000000G	EF	D0	00062		0903
00000000'	EF	00000000G	EF	D0	0006D		0904
00000000'	EF	00000000G	EF	D0	00078		0905
	08	AE	D4	00083	CLRL	DOING_BOLD	0906
59	01	7D	00086	MOVQ	#1, OPEN_QUOTE	:	0910
	57	7C	00089	CLRQ	DOING_UND	:	0907
	5B	D5	0008B	TSTL	LEN	:	0912
	03	14	0008D	BGTR	10\$	:	
	0612	31	0008F	BRW	91\$	:	
56	04	BE	9A	00092	MOVZBL	@PTR, CH	0918
	04	AE	D6	00096	INCL	PTR	
	5B	D7	00099	DECL	LEN	:	0919
1C	56	D1	0009B	CMPL	CH, #28	:	0921
	03	13	0009E	BEQL	11\$	:	
	00B2	31	000A0	BRW	18\$	:	
50	04	BE	9A	000A3	MOVZBL	@PTR, FNC	0932
	04	AE	D6	000A7	INCL	PTR	
51	04	BE	9A	000AA	MOVZBL	@PTR, OP	0933
	04	AE	D6	000AE	INCL	PTR	
	5B	02	C2	000B1	SUBL2	#2, LEN	0934
0000004F	8F	50	D1	000B4	CMPL	FNC, #79	0939
		6C	12	000BB	BNEQ	13\$	
00000000G	FF	51	90	000BD	MOVB	OP, @LP	0944
		EF	D6	000C4	INCL	LP	
		EF	D6	000CA	INCL	INTLIN	
		EF	D6	000D0	INCL	EXTLIN	
36 00000000G	EF	01	E1	000D6	BBC	#1, CMDBLK, 12\$	0946
	7E	04	AC	7D	000DE	MOVQ	TXI_LEN, -(SP)
		02	DD	000E2	PUSHL	#2	
		8F	DD	000E4	PUSHL	#DSRTOCS_TEXTD	
		7E	D4	000EA	CLRL	-(SP)	
		8F	DD	000EC	PUSHL	#DSRTOCS_OVERSTRK	
00000000G	00	06	FB	000F2	CALLS	#6, LIB\$SIGNAL	
00000000G	FF	EF	D0	000F9	MOVL	P.AAA, @LP	0961
00000000G	EF	04	C0	00104	ADDL2	#4, LP	



	00000000G	EF	04	C0	0010B	ADDL2	#4, INTLIN	...	0946
			3E	11	00112	BRB	17\$	...	0964
	00000000G	FF	25	90	00114	12\$:	MOVB	#37, @LP	
		00000000G	EF	D6	0011B	INCL	LP		
		00000000G	EF	D6	00121	INCL	INTLIN		
			29	11	00127	BRB	17\$	...	0936
	00000042	8F	50	D1	00129	13\$:	CMPL	FNC, #66	0968
			0C	12	00130	BNEQ	15\$		
		05	6E	E9	00132	BLBC	KEEP BOLD, 14\$	...	0972
		5A	01	D0	00135	MOVL	#1, BOLD_CHAR		
			18	11	00138	BRB	17\$		
			5A	D4	0013A	14\$:	CLRL	BOLD_CHAR	
			14	11	0013C	BRB	17\$		
	00000055	8F	50	D1	0013E	15\$:	CMPL	FNC, #85	0974
			0B	12	00145	BNEQ	17\$		
		05	AE	E9	00147	BLBC	KEEP UND, 16\$	...	0978
		58	01	D0	0014B	MOVL	#1, UND_CHAR		
			02	11	0014E	BRB	17\$		
			58	D4	00150	16\$:	CLRL	UND_CHAR	
			FF	36	31	00152	17\$:	BRW	9\$
		03	5A	E9	00155	18\$:	BLBC	BOLD_CHAR, 19\$	0994
			00C3	31	00158	BRW	31\$		
		03	08	AE	E8	0015B	19\$:	BLBS	DOING_BOLD, 21\$
			0101	31	0015F	20\$:	BRW	35\$	1001
		20	56	D1	00162	21\$:	CMPL	CH, #32	
			F8	13	00165	BEQL	20\$		
		50	00000000G	EF	D0	00167	MOVL	LP, R0	1009
51	00000000G	EF	01	EF	0016E	EXTZV	#1, #1, CMDBLK, R1		
			19	51	E9	00177	BLBC	R1, 22\$	
60		18	00	00000000'	EF	F0	0017A	INSV	P.AAB, #0, #24, (R0)
	00000000G	EF	03	C0	00183	ADDL2	#3, LP		
	00000000G	EF	03	C0	0018A	ADDL2	#3, INTLIN		
			15	11	00191	BRB	23\$		
		60	00000000'	EF	B0	00193	22\$:	MOVW	P.AAC, (R0)
	00000000G	EF	02	C0	0019A	ADDL2	#2, LP		
	00000000G	EF	02	C0	001A1	ADDL2	#2, INTLIN		
		57	57	E9	001A8	23\$:	BLBC	DOING_UND, 29\$	1011
		50	00000000G	EF	D0	001AB	MOVL	LP, R0	1019
		09	51	E9	001B2	BLBC	R1, 24\$		
		60	00000000'	EF	B0	001B5	MOVW	P.AAD, (R0)	
			07	11	001BC	BRB	25\$		
		60	00000000'	EF	B0	001BE	24\$:	MOVW	P.AAE, (R0)
	00000000G	EF	02	C0	001C5	25\$:	ADDL2	#2, LP	
	00000000G	EF	02	C0	001CC	ADDL2	#2, INTLIN		
		2A	58	E9	001D3	BLBC	UND_CHAR, 28\$	...	1021
		50	00000000G	EF	D0	001D6	MOVL	LP, R0	1029
		09	51	E9	001DD	BLBC	R1, 26\$		
		60	00000000'	EF	B0	001E0	MOVW	P.AAF, (R0)	
			07	11	001E7	BRB	27\$		
		60	00000000'	EF	B0	001E9	26\$:	MOVW	P.AAG, (R0)
	00000000G	EF	02	C0	001F0	27\$:	ADDL2	#2, LP	
	00000000G	EF	02	C0	001F7	ADDL2	#2, INTLIN		
			02	11	001FE	BRB	29\$		
			57	D4	00200	28\$:	CLRL	DOING_UND	1021
		14	51	E9	00202	29\$:	BLBC	R1, 30\$	1037
	00000000G	FF	8F	90	00205	MOVW	#93, @LP	...	1041
			00000000G	EF	D6	0020D	INCL	LP	



60

	00000000G	EF	D6	00213	INCL	INTLIN	
	08	AE	D4	00219	CLRL	DOING_BOLD	1043
		45	11	0021C	BRB	35\$	0994
	3F 08	AE	E8	0021E	31\$: BLBS	DOING_BOLD, 34\$	1053
17	00000000G	EF	D0	00222	MOVL	LP, R0	1060
	00000000G	EF	01	00229	BBC	#1, CMDBLK, 32\$	
	00000000G	EF	D0	00231	MOVL	P.AAH, (R0)	
	00000000G	EF	04	00238	ADDL2	#4, LP	
		EF	04	0023F	ADDL2	#4, INTLIN	
		15	11	00246	BRB	33\$	
	60 00000000'	EF	B0	00248	32\$: MOVW	P.AAI, (R0)	
	00000000G	EF	02	0024F	ADDL2	#2, LP	
	00000000G	EF	02	00256	ADDL2	#2, INTLIN	
	08	AE	01	0025D	33\$: MOVL	#1, DOING_BOLD	1062
			5A	00261	34\$: CLRL	BOLD_CHAR	1065
	03		58	00263	35\$: BLBC	UND_CHAR, 36\$	1068
		0082	31	00266	BRW	42\$	
	7D		57	00269	36\$: BLBC	DOING_UND, 41\$	1075
	20		56	0026C	CMPL	CH, #32	
			78	0026F	BEQL	41\$	
19	00000000G	EF	D0	00271	MOVL	LP, R0	1083
18	00000000'	EF	01	00278	BBC	#1, CMDBLK, 37\$	
	00000000G	EF	F0	00280	INSV	P.AAJ, #0, #24, (R0)	
	00000000G	EF	03	00289	ADDL2	#3, LP	
		EF	03	00290	ADDL2	#3, INTLIN	
		15	11	00297	BRB	38\$	
	60 00000000'	EF	B0	00299	37\$: MOVW	P.AAK, (R0)	
	00000000G	EF	02	002A0	ADDL2	#2, LP	
	00000000G	EF	02	002A7	ADDL2	#2, INTLIN	
31	00000000G	EF	01	002AE	38\$: BBC	#1, CMDBLK, 40\$	1085
	08	AE	E9	002B6	BLBC	DOING_BOLD, 39\$	1093
	00000000G	FF	D0	002BA	MOVL	P.AAL, @LP	
	00000000G	EF	04	002C5	ADDL2	#4, LP	
	00000000G	EF	04	002CC	ADDL2	#4, INTLIN	
	00000000G	FF	8F	90 002D3	39\$: MOVW	#93, @LP	1095
	00000000G	EF	D6	002DB	INCL	LP	
	00000000G	EF	D6	002E1	INCL	INTLIN	
		57	D4	002E7	40\$: CLRL	DOING_UND	1098
		43	11	002E9	41\$: BRB	46\$	1068
	3E		57	002EB	42\$: BLBS	DOING_UND, 45\$	1108
17	00000000G	EF	D0	002EE	MOVL	LP, R0	1115
	00000000'	EF	01	002F5	BBC	#1, CMDBLK, 43\$	
	00000000G	EF	D0	002FD	MOVL	P.AAM, (R0)	
	00000000G	EF	04	00304	ADDL2	#4, LP	
	00000000G	EF	04	0030B	ADDL2	#4, INTLIN	
		15	11	00312	BRB	44\$	
	60 00000000'	EF	B0	00314	43\$: MOVW	P.AAN, (R0)	
	00000000G	EF	02	0031B	ADDL2	#2, LP	
	00000000G	EF	02	00322	ADDL2	#2, INTLIN	
		57	01	00329	44\$: MOVL	#1, DOING_UND	1117
			58	0032C	45\$: CLRL	UND_CHAR	1120
	20		56	0032E	46\$: CMPL	CH, #32	1126
			0C	12 00331	BNEQ	48\$	
			01	DD 00333	PUSHL	#1	
	00000000V	EF	01	FB 00335	CALLS	#1, ENDWRD	1127
		FD4C	31	0033C	47\$: BRW	9\$	
	3E		56	0033F	48\$: CMPL	CH, #62	1129

		03	13	00342	BEQL	49\$		
		010B	31	00344	BRW	54\$		
	50	00000000G	EF	D0 00347	49\$:	MOVL	LP, R0	1135
	03	00000000G	EF	D1 0034E		CMPL	MAJOR, #3	1132
			03	13 00355		BEQL	50\$	
			00E0	31 00357		BRW	52\$	
	60		56	90 0035A	50\$:	MOVB	CH, (R0)	1135
		00000000G	EF	D6 0035D		INCL	LP	
		00000000G	EF	D6 00363		INCL	INTLIN	
		00000000G	EF	D6 00369		INCL	EXTLIN	
C5	00000000G	EF	01	E1 0036F		BBC	#1, CMDBLK, 47\$	1137
	10	AE	00000000G	EF	B0 00377	MOVW	INTLIN, \$STR\$STRING	1143
	12	AE		0E	90 0037F	MOVB	#14, \$STR\$STRING+2	
	13	AE		01	90 00383	MOVB	#1, \$STR\$STRING+3	
	14	AE	00000000G	EF	9E 00387	MOVAB	LINE, \$STR\$STRING+4	
				00000000G	EF	9F 0038F	PUSHAB	STR\$FAILURE
					7E	D4 00395	CLRL	-(SP)
		00000000G	EF	9F 00397		PUSHAB	\$STR\$TARGET	
		1C	AE	9F 0039D		PUSHAB	\$STR\$STRING	
			7E	D4 003A0		CLRL	-(SP)	
	00000000G	EF	05	FB 003A2		CALLS	#5, XST\$COPY	
	50	00000000G	EF	3C 003A9		MOVZWL	TMPSTR, R0	
	00000000G	EF	50	C0 003B0		ADDL2	R0, CHROUT	
	00000000G	EF	00000000G	EF	9E 003B7	MOVAB	\$IOB\$OUTPUT, IOB\$+68	
	00000000G	EF	07	90 003C2		MOVB	#7, IOB\$+44	
				00000000G	EF	9F 003C9	PUSHAB	XPO\$FAILURE
			7E	D4 003CF		CLRL	-(SP)	
		00000000G	EF	9F 003D1		PUSHAB	IOB\$	
	00000000G	EF	03	FB 003D7		CALLS	#3, XPO\$PUT	
07	00000000G	EF	01	E1 003DE		BBC	#1, CMDBLK, 51\$	
	00000000V	EF	00	FB 003E6		CALLS	#0, SPLIT	
	00000000G	EF	00000000G	EF	9E 003ED	51\$:	MOVAB	LINE, LP
				00000000G	EF	D4 003F8	CLRL	INTLIN
				00000000G	EF	D4 003FE	CLRL	EXTLIN
	00000000G	FF	3C	90 00404		MOVB	#60, @LP	1145
				00000000G	EF	D6 0040B	INCL	LP
				00000000G	EF	D6 00411	INCL	INTLIN
	000000000'	EF	00000000G	EF	D0 00417	MOVL	LP, WRDPTR	1146
	000000000'	EF	00000000G	EF	D0 00422	MOVL	EXTLIN, EXTWRD	1147
	000000000'	EF	00000000G	EF	D0 0042D	MOVL	INTLIN, INTWRD	1148
			59	11 00438		BRB	57\$	1132
0A	00000000G	EF	01	E1 0043A	52\$:	BBC	#1, CMDBLK, 53\$	1155
	60	000000000'	EF	B0 00442		MOVW	P.AAO, (R0)	1157
			01E4	31 00449		BRW	85\$	
	60		56	90 0044C	53\$:	MOVB	CH, (R0)	1159
			023D	31 0044F		BRW	89\$	
			51	D4 00452	54\$:	CLRL	R1	1165
	20		56	D1 00454		CMPL	CH, #32	
			02	18 00457		BGEQ	55\$	
			51	D6 00459		INCL	R1	
			50	D4 0045B	55\$:	CLRL	R0	
	0000007E	8F	56	D1 0045D		CMPL	CH, #126	
			02	15 00464		BLEQ	56\$	
			50	D6 00466		INCL	R0	
	50		51	C8 00468	56\$:	BISL2	R1, R0	
	01		50	D1 0046B		CMPL	R0, #1	
			59	12 0046E		BNEQ	59\$	



FORMAT  
V04-000

FORMAT - generate formatted output lines  
FMTTXT - scan and format text

M 14  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 25  
(3)

1D 00000000G	EF	01	E1	00470	BBC	#1, CMDBLK, 58\$	1171
	7E	04	AC	7D 00478	MOVQ	TXI_LEN, -(SP)	1179
			02	DD 0047C	PUSHL	#2	
	00000000G	8F	DD	0047E	PUSHL	#DSRTOCS_TEXTD	
		7E	D4	00484	CLRL	-(SP)	
	00000000G	8F	DD	00486	PUSHL	#DSRTOCS_CTRLCHAR	
00000000G	00	06	FB	0048C	CALLS	#6, LIB\$SIGNAL	
		6E	11	00493	BRB	62\$	1171
00000000G	FF	5F	8F	90 00495	MOVB	#95, @LP	1192
		00000000G	EF	D6 0049D	INCL	LP	
	00000000G	EF	D6	004A3	INCL	INTLIN	
00000000G	FF	56	90	004A9	MOVB	CH, @LP	1193
		00000000G	EF	D6 004B0	INCL	LP	
	00000000G	EF	D6	004B6	INCL	INTLIN	
	08	56	D1	004BC	CMPL	CH, #8	1195
		42	12	004BF	BNEQ	62\$	
	00000000G	EF	D7	004C1	DECL	EXTLIN	1197
		3A	11	004C7	BRB	62\$	1123
	03 00000000G	EF	D1	004C9	CMPL	MAJOR, #3	1209
		03	12	004D0	BNEQ	60\$	
		01B3	31	004D2	BRW	88\$	
03 00000000G	EF	01	E0	004D5	BBS	#1, CMDBLK, 61\$	1223
		0160	31	004DD	BRW	86\$	
0000005F	8F	56	D1	004E0	CMPL	CH, #95	1233
		1D	12	004E7	BNEQ	63\$	
00000000G	FF	05	28	004E9	MOV C3	#5, P.AAP, @LP	1234
	00000000G	05	C0	004F5	ADDL2	#5, LP	
	00000000G	05	C0	004FC	ADDL2	#5, INTLIN	
		FB85	31	00503	BRW	9\$	1230
	2D	56	D1	00506	CMPL	CH, #45	1236
		0D	12	00509	BNEQ	64\$	
00000000G	FF	00000000'	EF	B0 0050B	MOVW	P.AAQ, @LP	1237
		72	11	00516	BRB	70\$	
	2A	56	D1	00518	CMPL	CH, #42	1239
		0D	12	0051B	BNEQ	65\$	
00000000G	FF	00000000'	EF	B0 0051D	MOVW	P.AAR, @LP	1240
		76	11	00528	BRB	72\$	
	3D	56	D1	0052A	CMPL	CH, #61	1242
		0D	12	0052D	BNEQ	66\$	
00000000G	FF	00000000'	EF	B0 0052F	MOVW	P.AAS, @LP	1243
		7A	11	0053A	BRB	74\$	
	2B	56	D1	0053C	CMPL	CH, #43	1245
		0D	12	0053F	BNEQ	67\$	
00000000G	FF	00000000'	EF	B0 00541	MOVW	P.AAT, @LP	1246
		7E	11	0054C	BRB	76\$	
0000005C	8F	56	D1	0054E	CMPL	CH, #92	1248
		0D	12	00555	BNEQ	68\$	
00000000G	FF	00000000'	EF	B0 00557	MOVW	P.AAU, @LP	1249
		7A	11	00562	BRB	78\$	
00000040	8F	56	D1	00564	CMPL	CH, #64	1251
		0D	12	0056B	BNEQ	69\$	
00000000G	FF	00000000'	EF	B0 0056D	MOVW	P.AAV, @LP	1252
		7A	11	00578	BRB	80\$	
	2F	56	D1	0057A	CMPL	CH, #47	1254
		0D	12	0057D	BNEQ	71\$	
00000000G	FF	00000000'	EF	B0 0057F	MOVW	P.AAW, @LP	1255
		7E	11	0058A	BRB	82\$	

FORMAT  
V04-000

FORMAT - generate formatted output lines  
FMTTXT - scan and format text

N 14  
16-Sep-1984 00:34:26 VAX-11 Bliss-32 V4.0-742  
14-Sep-1984 13:06:27 [RUNOFF.SRC]FORMAT.BLI;1

Page 26  
(3)

0000007C	8F	56	D1	0058C	71\$:	CMPL	CH, #124	1257
		0D	12	00593		BNEQ	73\$	
00000000G	FF 00000000'	EF	B0	00595		MOVW	P.AAX, @LP	1258
		68	11	005A0	72\$:	BRB	82\$	
0000007B	8F	56	D1	005A2	73\$:	CMPL	CH, #123	1260
		0D	12	005A9		BNEQ	75\$	
00000000G	FF 00000000'	EF	B0	005AB		MOVW	P.AAY, @LP	1261
		78	11	005B6	74\$:	BRB	85\$	
0000007D	8F	56	D1	005B8	75\$:	CMPL	CH, #125	1263
		0D	12	005BF		BNEQ	77\$	
00000000G	FF 00000000'	EF	B0	005C1		MOVW	P.AAZ, @LP	1264
		62	11	005CC	76\$:	BRB	85\$	
	3C	56	D1	005CE	77\$:	CMPL	CH, #60	1266
		0D	12	005D1		BNEQ	79\$	
00000000G	FF 00000000'	EF	B0	005D3		MOVW	P.ABA, @LP	1267
		50	11	005DE	78\$:	BRB	85\$	
0000005B	8F	56	D1	005E0	79\$:	CMPL	CH, #91	1269
		0D	12	005E7		BNEQ	81\$	
00000000G	FF 00000000'	EF	B0	005E9		MOVW	P.ABB, @LP	1270
		3A	11	005F4	80\$:	BRB	85\$	
0000005D	8F	56	D1	005F6	81\$:	CMPL	CH, #93	1272
		0D	12	005FD		BNEQ	83\$	
00000000G	FF 00000000'	EF	B0	005FF		MOVW	P.ABC, @LP	1273
		24	11	0060A	82\$:	BRB	85\$	
	22	56	D1	0060C	83\$:	CMPL	CH, #34	1275
		77	12	0060F		BNEQ	88\$	
	50 00000000G	EF	D0	00611		MOVL	LP, R0	1284
	0B	59	E9	00618		BLBC	OPEN_QUOTE, 84\$	1278
	60 00000000'	EF	B0	0061B		MOVW	P.ABD, (R0)	1284
		59	D4	00622		CLRL	OPEN_QUOTE	1285
		0A	11	00624		BRB	85\$	1278
	60 00000000'	EF	B0	00626	84\$:	MOVW	P.ABE, (R0)	1292
		01	D0	0062D		MOVL	#1, OPEN_QUOTE	1293
00000000G	EF	02	C0	00630	85\$:	ADDL2	#2, LP	1284
00000000G	EF	02	C0	00637		ADDL2	#2, INTLIN	
		61	11	0063E		BRB	90\$	1230
0000005F	8F	56	D1	00640	86\$:	CMPL	CH, #95	1312
		2B	13	00647		BEQL	87\$	
	2A	56	D1	00649		CMPL	CH, #42	1313
		26	13	0064C		BEQL	87\$	
	21	56	D1	0064E		CMPL	CH, #33	1314
		21	13	00651		BEQL	87\$	
	2E	56	D1	00653		CMPL	CH, #46	1315
		1C	13	00656		BEQL	87\$	
0000005C	8F	56	D1	00658		CMPL	CH, #92	1316
		13	13	0065F		BEQL	87\$	
	25	56	D1	00661		CMPL	CH, #37	1317
		0E	13	00664		BEQL	87\$	
	26	56	D1	00666		CMPL	CH, #38	1318
		09	13	00669		BEQL	87\$	
0000005E	8F	56	D1	0066B		CMPL	CH, #94	1319
		14	12	00672		BNEQ	88\$	
00000000G	FF 5F 00000000G	8F	90	00674	87\$:	MOVB	#95, @LP	1324
	00000000G	EF	D6	0067C		INCL	LP	
		EF	D6	00682		INCL	INTLIN	
00000000G	FF 00000000G	56	90	00688	88\$:	MOVB	CH, @LP	1326
		EF	D6	0068F	89\$:	INCL	LP	

GBLI  
V04-



FORMAT  
V04-000

FORMAT - generate formatted output lines  
FMTTXT - scan and format text

B 15  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 27  
(3)

GBL  
V04

			00000000G	EF	D6	00695	INCL	INTLIN	:	
			00000000G	EF	D6	0069B	INCL	EXTLIN	:	
				F9E7	31	006A1	90\$:	BRW	9\$	1123
				7E	D4	006A4	91\$:	CLRL	-(SP)	1338
				01	FB	006A6		CALLS	#1, ENDWRD	
				01	E1	006AD		BBC	#1, CMDBLK, 95\$	1340
				03	AE	E8 006B5		BLBS	DOING_BOLD, 92\$	1344
				4F	E9	006B9		BLBC	DOING_UND, 94\$	
				00	FO	006BC	92\$:	INSV	P.ABF, #0, #24, @LP	1350
				03	CO	006C9		ADDL2	#3, LP	
				03	CO	006D0		ADDL2	#3, INTLIN	
				1C	AE	E9 006D7		BLBC	DOING_BOLD, 93\$	1355
				19	E9	006DB		BLBC	DOING_UND, 93\$	
				FF	BO	006DE		MOVW	P.ABG, @LP	
				02	CO	006E9		ADDL2	#2, LP	
				02	CO	006F0		ADDL2	#2, INTLIN	
				5D	8F	90 006F7	93\$:	MOVB	#93, @LP	1357
				00000000G	EF	D6 006FF		INCL	LP	
				00000000G	EF	D6 00705		INCL	INTLIN	
				55	E8	0070B	94\$:	BLBS	OPEN QUOTE, 97\$	1360
				7E	AC	7D 0070E		MOVQ	TXT_LEN, -(SP)	1367
					02	DD 00712		PUSHL	#2	
					8F	DD 00714		PUSHL	#DSRTOCS_TEXTD	
					7E	D4 0071A		CLRL	-(SP)	
					8F	DD 0071C		PUSHL	#DSRTOCS_CLOSEQUOT	
					06	FB 00722		CALLS	#6, LIB\$SIGNAL	
					04	00729		RET		1340
					AE	E9 0072A	95\$:	BLBC	DOING_BOLD, 96\$	1381
					EF	BO 0072E		MOVW	P.ABH, @LP	
					02	CO 00739		ADDL2	#2, LP	
					02	CO 00740		ADDL2	#2, INTLIN	
					57	E9 00747	96\$:	BLBC	DOING_UND, 97\$	1383
					EF	BO 0074A		MOVW	P.ABI, @LP	
					02	CO 00755		ADDL2	#2, LP	
					02	CO 0075C		ADDL2	#2, INTLIN	
					04	00763	97\$:	RET		1387

; Routine Size: 1892 bytes, Routine Base: \$CODE\$ + 00A3



```

: 916 1388 1 %SBTTL 'ENDWRD - verify word fits on line'
: 917 1389 1 ROUTINE endwr (space) : NOVALUE =
: 918 1390 1 ++
: 919 1391 1 FUNCTIONAL DESCRIPTION:
: 920 1392 1
: 921 1393 1     ENDWRD is called when a space is about to be output. For RUNOFF,
: 922 1394 1     it makes sure that the word that the space ends fits on the the line.
: 923 1395 1     If it doesn't, it wraps the line.
: 924 1396 1
: 925 1397 1 FORMAL PARAMETERS:
: 926 1398 1
: 927 1399 1     space          - true if a space is to be generated
: 928 1400 1
: 929 1401 1 IMPLICIT INPUTS:
: 930 1402 1
: 931 1403 1     cmdblk          - command line information block
: 932 1404 1     rmargin         - indicates how far to the right this word can extend.
: 933 1405 1     line_indent    - number of columns of indentation
: 934 1406 1     wrap           - first column to start new line in
: 935 1407 1     wrdptr         - pointer to beginning of word
: 936 1408 1     extwr         - external length of line not including word
: 937 1409 1     intwr         - internal length of line not including word
: 938 1410 1
: 939 1411 1 IMPLICIT OUTPUTS:
: 940 1412 1
: 941 1413 1     wrdptr         - points to the end of the current word
: 942 1414 1     extwr         - new external line length
: 943 1415 1     intwr         - new internal line length
: 944 1416 1
: 945 1417 1 ROUTINE VALUE:
: 946 1418 1 COMPLETION CODES:
: 947 1419 1
: 948 1420 1     NONE
: 949 1421 1
: 950 1422 1 SIDE EFFECTS:
: 951 1423 1
: 952 1424 1     NONE
: 953 1425 1
: 954 1426 1 --
: 955 1427 1
: 956 1428 1 BEGIN
: 957 1429 1
: 958 1430 1 IF (.extlin GTR .rmargin) AND ( NOT .cmdblk [contents$v_tms11])
: 959 1431 1 THEN
: 960 1432 1     BEGIN
: 961 1433 1
: 962 1434 1     The word that this space terminates does not fit.  Wrap the line.
: 963 1435 1     First determine the length of the word just ended.
: 964 1436 1     Note that WORD_xxxxx were set at the beginning of the word being
: 965 1437 1     terminated, while the normal counters have been updated ever since.
: 966 1438 1
: 967 1439 1     extwr = .extlin - .extwr;
: 968 1440 1     intwr = .intlin - .intwr;
: 969 1441 1
: 970 1442 1     Now adjust the current line lengths before outputting the line
: 971 1443 1
: 972 1444 1     extlin = .extlin - .extwr;

```



```

: 973      1445      | intlin = .intlin - .intwr;
: 974      1446      |
: 975      1447      |   Before outputting the line that is to be wrapped, make sure that
: 976      1448      |   at least two lines are still available on the page. This avoids
: 977      1449      |   having the first part of the text on one page and the last part
: 978      1450      |   of it on another page.
: 979      1451      |
: 980      1452      | put ('.TEST PAGE 2');
: 981      1453      |
: 982      1454      |   And now output the line, up to but not including the word that
: 983      1455      |   this space terminates.
: 984      1456      |
: 985      1457      | put ((.intlin, CH$PTR (line)));
: 986      1458      | clr_line ();
: 987      1459      |
: 988      1460      |   Add sufficient spaces to align the wrapped word with the first
: 989      1461      |   character of the line that was just terminated.
: 990      1462      |
: 991      1463      | pad ((.wrap - .line_indent));
: 992      1464      |
: 993      1465      |   Adjust the external line length.
: 994      1466      |   It really represents .line_indent additional characters.
: 995      1467      |
: 996      1468      | extlin = .extlin + .line_indent;
: 997      1469      |
: 998      1470      |   At this point the word that would have overflowed the line is
: 999      1471      |   sitting out in limbo. But, we know its length and where it is.
1000      1472      |   Move it to the left so that it's aligned properly.
1001      1473      |
1002      1474      |
1003      1475      | INCR i FROM 1 TO .intwr DO
1004      1476      |     CH$WCHAR_A (CH$RCHAR_A (wrptr), lp);
1005      1477      |
1006      1478      |
1007      1479      |   And finally, update the counters that were bypassed in the move
1008      1480      |
1009      1481      | extlin = .extlin + .extwr;
1010      1482      | intlin = .intlin + .intwr;
1011      1483      | END;
1012      1484      |
1013      1485      | IF .space THEN write_char ('%', counts_visually);
1014      1486      |
1015      1487      |
1016      1488      |   Remember current lengths for use the next time around.
1017      1489      |
1018      1490      | extwr = .extlin;
1019      1491      | intwr = .intlin;
1020      1492      | wrptr = .lp;
1021      1493      | END;

```

! End of endwr

```

                                .PSECT $PLITS,NOWRT,NOEXE,2
32 20 45 47 41 50 20 54 53 45 54 2E 00090 P.ABJ: .ASCII \.TEST PAGE 2\
                                .PSECT $OWNS,NOEXE,2

```



			OFFC	00000	ENDWRD:	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 1389
	5B	00000000G	EF	9E	00002	MOVAB	LP, R11	:
	5A	00000000G	EF	9E	00009	MOVAB	INTLIN, R10	:
	59	00000000G	EF	9E	00010	MOVAB	EXTLIN, R9	:
	58	00000000'	EF	9E	00017	MOVAB	INTWRD, R8	:
	5E		08	C2	0001E	SUBL2	#8, SP	:
	00000000G		69	D1	00021	CMPL	EXTLIN, RMARGIN	: 1430
			03	14	00028	BGTR	2\$	:
			0137	31	0002A	BRW	8\$	:
F5	00000000G		01	E0	0002D	BBS	#1, CMDBLK, 1\$	:
A8		FC	A8	C3	00035	SUBL3	EXTWRD, EXTLIN, EXTWRD	: 1439
68			68	C3	0003B	SUBL3	INTWRD, INTLIN, INTWRD	: 1440
		FC	A8	C2	0003F	SUBL2	EXTWRD, EXTLIN	: 1444
			68	C2	00043	SUBL2	INTWRD, INTLIN	: 1445
	00000000G		EF	9F	00046	PUSHAB	STR\$FAILURE	: 1452
			7E	D4	0004C	CLRL	-(SP)	:
	00000000G		EF	9F	0004E	PUSHAB	\$STR\$TARGET	:
		04	A8	9F	00054	PUSHAB	\$STR\$STRING	:
			7E	D4	00057	CLRL	-(SP)	:
	00000000G		EF	FB	00059	CALLS	#5, XST\$COPY	:
			50	3C	00060	MOVZWL	TMPSTR, R0	:
	00000000G		EF	C0	00067	ADDL2	R0, CHROUT	:
	00000000G		EF	9E	0006E	MOVAB	\$IOB\$OUTPUT, IOB\$+68	:
	00000000G		EF	90	00079	MOVB	#7, IOB\$+44	:
			EF	9F	00080	PUSHAB	XPO\$FAILURE	:
			7E	D4	00086	CLRL	-(SP)	:
	00000000G		EF	9F	00088	PUSHAB	IOB\$	:
			03	FB	0008E	CALLS	#3, XPO\$PUT	:
07	00000000G		01	E1	00095	BBC	#1, CMDBLK, 3\$	:
	00000000V		00	FB	0009D	CALLS	#0, SPLIT	:
			6A	B0	000A4	MOVW	INTLIN, \$STR\$STRING	: 1457
			0E	90	000A7	MOVB	#14, \$STR\$STRING+2	:
	02	AE	01	90	000AB	MOVB	#1, \$STR\$STRING+3	:
	03	AE	EF	9E	000AF	MOVAB	LINE, \$STR\$STRING+4	:
	04	AE	EF	9F	000B7	PUSHAB	STR\$FAILURE	:
			7E	D4	000BD	CLRL	-(SP)	:
	00000000G		EF	9F	000BF	PUSHAB	\$STR\$TARGET	:
		0C	AE	9F	000C5	PUSHAB	\$STR\$STRING	:
			7E	D4	000C8	CLRL	-(SP)	:
	00000000G		EF	FB	000CA	CALLS	#5, XST\$COPY	:
			50	3C	000D1	MOVZWL	TMPSTR, R0	:
	00000000G		EF	C0	000D8	ADDL2	R0, CHROUT	:
	00000000G		EF	9E	000DF	MOVAB	\$IOB\$OUTPUT, IOB\$+68	:
	00000000G		EF	90	000EA	MOVB	#7, IOB\$+44	:
			EF	9F	000F1	PUSHAB	XPO\$FAILURE	:
			7E	D4	000F7	CLRL	-(SP)	:
	00000000G		EF	9F	000F9	PUSHAB	IOB\$	:



FORMAT  
V04-000

FORMAT - generate formatted output lines  
ENDWRD - verify word fits on line

F 15  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 31  
(4)

GCO

	07	00000000G	EF		03	FB	000FF	CALLS	#3, XPO\$PUT		
		00000000G	EF		01	E1	00106	BBC	#1, CMDBLK, 4\$		
		00000000V	EF		00	FB	0010E	CALLS	#0, SPLIT		
			6B	00000000G	EF	9E	00115	MOVAB	LINE, LP	1458	
					6A	D4	0011C	CLRL	INTLIN		
					69	D4	0011E	CLRL	EXTLIN		
			57	00000000G	EF	D0	00120	MOVL	LINE, INDENT, R7	1463	
			57	00000000G	EF	D1	00127	CMPL	WRAP, R7		
					18	15	0012E	BLEQ	5\$		
56	56	00000000G	EF		57	C3	00130	SUBL3	R7, WRAP, R6		
	20		6E		00	2C	00138	MOVCS	#0, (SP), #32, R6, @LP		
				00	BB		0013D				
			6B		56	C0	0013F	ADDL2	R6, LP		
			6A		56	C0	00142	ADDL2	R6, INTLIN		
			69		56	C0	00145	ADDL2	R6, EXTLIN		
			69		57	C0	00148	ADDL2	R7, EXTLIN	1468	
					50	D4	0014B	CLRL	I	1476	
					0A	11	0014D	BRB	7\$		
		00	BB	F8	BB	90	0014F	MOVBS	@WRDPTR, @LP		
				F8	A8	D6	00154	INCL	WRDPTR		
					6B	D6	00157	INCL	LP		
	F2		50		68	F3	00159	AOBLEQ	INTWRD, I, 6\$		
			69	FC	A8	C0	0015D	ADDL2	EXTWRD, EXTLIN	1481	
			6A		68	C0	00161	ADDL2	INTWRD, INTLIN	1482	
			0A	04	AC	E9	00164	BLBC	SPACE, 9\$	1485	
		00	BB		20	90	00168	MOVBS	#32, @LP		
					6B	D6	0016C	INCL	LP		
					6A	D6	0016E	INCL	INTLIN		
					69	D6	00170	INCL	EXTLIN		
		FC	A8		69	D0	00172	MOVL	EXTLIN, EXTWRD	1490	
			68		6A	D0	00176	MOVL	INTLIN, INTWRD	1491	
		F8	A8		6B	D0	00179	MOVL	LP, WRDPTR	1492	
					04	0017D		RET		1493	

; Routine Size: 382 bytes, Routine Base: \$CODE\$ + 0807



FORMAT  
V04-000

FORMAT - generate formatted output lines  
SPLIT - start new output file for tms if necess

G 15  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 32  
(5)

```

1023 1494 1 %SBTTL 'SPLIT - start new output file for tms if necessary'
1024 1495 1 GLOBAL ROUTINE split : NOVALUE =
1025 1496 1 ++
1026 1497 1
1027 1498 1 FUNCTIONAL DESCRIPTION:
1028 1499 1
1029 1500 1 This routine checks to see if the TMS output must be split
1030 1501 1 to another output file. This is necessary to prevent long
1031 1502 1 galleys which could jam the typesetter.
1032 1503 1
1033 1504 1 FORMAL PARAMETERS:
1034 1505 1
1035 1506 1 None
1036 1507 1
1037 1508 1 IMPLICIT INPUTS:
1038 1509 1
1039 1510 1 chrout - number of characters written to the current output file.
1040 1511 1
1041 1512 1 IMPLICIT OUTPUTS:
1042 1513 1
1043 1514 1 None
1044 1515 1
1045 1516 1 ROUTINE VALUE:
1046 1517 1 COMPLETION CODES:
1047 1518 1
1048 1519 1 None
1049 1520 1
1050 1521 1 SIDE EFFECTS:
1051 1522 1
1052 1523 1 None
1053 1524 1 --
1054 1525 1
1055 1526 1 BEGIN
1056 1527 1
1057 1528 1 IF .chrout GEQ tms_characters_per_file
1058 1529 1 THEN
1059 1530 1 BEGIN
1060 1531 1
1061 1532 1 Must start a new output file
1062 1533 1
1063 1534 1 LOCAL
1064 1535 1
1065 1536 1 name_len,
1066 1537 1 spec_blk : $xpo_spec_block;
1067 1538 1
1068 1539 1 IF .outfile [str$h_length] EQL 0
1069 1540 1 THEN
1070 1541 1
1071 1542 1 Save current output file name
1072 1543 1
1073 1544 1 $str_copy (string = tocoob [iob$t_resultant], target = outfile);
1074 1545 1
1075 1546 1
1076 1547 1 Write terminator to current file and close it
1077 1548 1
1078 1549 1 $xpo_put (iob = tocoob, string = '*cfini*');
1079 1550 1 $xpo_close (iob = tocoob);
```



```

: 1080      1551      |
: 1081      1552      | Compute new file name
: 1082      1553      |
: 1083      1554      | $xpo_parse_spec (file_spec = outfile, spec_block = spec_blk);
: 1084      1555      | name_len = (IF .spec_blk [xpo$h_file_name] GEQ 6 THEN 6 ELSE .spec_blk [xpo$h_file_name]);
: 1085      1556      | fileno = .fileno + 1;
: 1086      1557      |
: 1087      1558      | Initialize IOB, open new file and reset character count
: 1088      1559      |
: 1089      1560      | $xpo_iob_init (iob = tocoob);
: 1090      1561      | $xpo_open (iob = tocoob, options = output, default = outfile,
: 1091      1562      |   file_spec = $str_concat ((.name_len, .spec_blk [xpo$a_file_name]),
: 1092      1563      |   $str_ascii (.fileno, UNSIGNED, leading_zero, length = 3))
: 1093      1564      |   %IF %BLISS (BLISS32) %THEN , failure = open_error %FI
: 1094      1565      | );
: 1095      1566      | chrout = 0;
: 1096      1567      |
: 1097      1568      | Tell user about new file
: 1098      1569      |
: 1099      1570      |
: 1100      1571      | %IF %BLISS (BLISS32)
: 1101      1572      | %THEN
: 1102      1573      |   SIGNAL (contents$_tms11, 1, tocoob [iob$t_resultant]);
: 1103      1574      | %ELSE
: 1104      1575      |   $xpo_put_msg (severity = success,
: 1105      1576      |   string = $str_concat ('output file full - continuing with file ',
: 1106      1577      |   tocoob [iob$t_resultant], ''));
: 1107      1578      | %FI
: 1108      1579      |
: 1109      1580      | Write file prologue
: 1110      1581      |
: 1111      1582      |
: 1112      1583      | put ('*start*');
: 1113      1584      | put ('*cinit*');
: 1114      1585      | END;
: 1115      1586      |
: 1116      1587      | END;

```

```

                                .PSECT  $SPLIT$,NOWRT,NOEXE,2
2A 69 6E 69 66 63 2A 0009C P.ABK: .ASCII  \*cfini*\
2A 74 72 61 74 73 2A 000A3 P.ABL: .ASCII  \*start*\
2A 74 69 6E 69 63 2A 000AA P.ABM: .ASCII  \*cinit*\

                                .PSECT  $OWNS$,NOEXE,2
0007 00020 $IOB$OUTPUT:
                                .WORD  7
01 0E 00022 .BYTE 14, 1
00000000' 00024 .ADDRESS P.ABK
0007 00028 $STR$STRING:
                                .WORD  7
01 0E 0002A .BYTE 14, 1
00000000' 0002C .ADDRESS P.ABL
0007 00030 $STR$STRING:

```



01 0E 00032  
00000000' 00034

.WORD 7  
.BYTE 14, 1  
.ADDRESS P.ABM

\$STR\$TARGET=  
\$STR\$FILE\_SPEC=  
\$IOB\$DEFAULT=  
.EXTRN XPOS\$CLOSE, XPOS\$PARSE\_SPEC  
.EXTRN XST\$JOIN, XST\$ASCII  
.EXTRN XPOS\$OPEN

.PSECT \$CODE\$,NOWRT,2

			OFFC 00000	.ENTRY SPLIT, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 ; 1495
	5B 00000000G	EF 9E 0C002	MOVAB CHROUT, R11	
	5A 00000000G	EF 9E 00009	MOVAB XPOS\$FAILURE, R10	
	59 00000000G	EF 9E 00010	MOVAB \$STR\$TARGET, R9	
	58 00000000'	EF 9E 00017	MOVAB OUTFILE, R8	
	57 00000000G	EF 9E 0001E	MOVAB IOB\$, R7	
	5E B0	AE 9E 00025	MOVAB -80(SP), SP	
00002800	8F	6B D1 00029	CMPL CHROUT, #10240	1528
		01 18 00030	BGEQ 1\$	
		04 00032	RET	
		68 B5 00033 1\$:	TSTW OUTFILE	1539
		16 12 00035	BNEQ 2\$	
	00000000G	EF 9F 00037	PUSHAB STR\$FAILURE	1544
		7E D4 0003D	CLRL -(SP)	
		58 DD 0003F	PUSHL R8	
	1C	A7 9F 00041	PUSHAB \$STR\$STRING	
		7E D4 00044	CLRL -(SP)	
00000000G	EF	05 FB 00046	CALLS #5, XST\$COPY	
44	A7	A8 9E 0004D 2\$:	MOVAB \$IOB\$OUTPUT, IOB\$+68	1549
2C	A7	07 90 00052	MOVB #7, IOB\$+44	
		5A DD 00056	PUSHL R10	
		7E D4 00058	CLRL -(SP)	
		57 DD 0005A	PUSHL R7	
00000000G	EF	03 FB 0005C	CALLS #3, XPOS\$PUT	
2C	A7	02 90 00063	MOVB #2, IOB\$+44	1550
		5A DD 00067	PUSHL R10	
		7E D4 00069	CLRL -(SP)	
		57 DD 0006B	PUSHL R7	
00000000G	EF	03 FB 0006D	CALLS #3, XPOS\$CLOSE	
		5A DD 00074	PUSHL R10	1554
		7E D4 00076	CLRL -(SP)	
	7E	01 CE 00078	MNEGL #1, -(SP)	
		AE 9F 0007B	PUSHAB SPEC_BLK	
	14	58 DD 0007E	PUSHL R8	
00000000G	EF	05 FB 00080	CALLS #5, XPOS\$PARSE_SPEC	
06		AE B1 00087	CMPL SPEC_BLK+32, #6	1555
		05 1F 0008B	BLSSU 3\$	
	56	06 D0 0008D	MOVL #6, NAME_LEN	
		04 11 00090	BRB 4\$	
	56	AE 3C 00092 3\$:	MOVZWL SPEC_BLK+32, NAME_LEN	
	28	A8 D6 00096 4\$:	INCL FILENO	1556
00F4 8F	00	00 2C 00099	MOVCS #0, (SP), #0, #244, IOB\$	1560
		67 000A0		
	67 0301003D	8F D0 000A1	MOVL #50397245, IOB\$	



1E	A7	020E	8F	B0	000A8	MOVW	#526, IOB\$RESULTANT+2	:	1565
		FC	03	DD	000AE	PUSHL	#3	:	
		0603	A8	DD	000B0	PUSHL	FILENO	:	
00000000G	7E		8F	3C	000B3	MOVZWL	#1539, -(SP)	:	
	EF		03	FB	000B8	CALLS	#3, XST\$ASCII	:	
	6E		56	B0	000BF	MOVW	NAME_LEN, \$STR\$STRINGO	:	
02	AE		0E	90	000C2	MOVB	#14, \$STR\$STRINGO+2	:	
03	AE		01	90	000C6	MOVB	#1, \$STR\$STRINGO+3	:	
04	AE	2C	AE	D0	000CA	MOVL	SPEC_BLK+36, \$STR\$STRINGO+4	:	
			50	DD	000CF	PUSHL	R0	:	
		04	AE	9F	000D1	PUSHAB	\$STR\$STRINGO	:	
00000000G	EF		02	FB	000D4	CALLS	#2, XST\$JOIN	:	
04	A7		50	D0	000DB	MOVL	R0, IOB\$+4	:	
08	A7		68	9E	000DF	MOVAB	\$IOB\$DEFAULT, IOB\$+8	:	
2E	A7		02	88	000E3	BISB2	#2, IOB\$+46	:	
2C	A7		01	90	000E7	MOVB	#1, IOB\$+44	:	
		00000000G	EF	9F	00CEB	PUSHAB	OPEN_ERROR	:	
			7E	D4	000F1	CLRL	-(SP)	:	
00000000G	EF		57	DD	000F3	PUSHL	R7	:	
			03	FB	000F5	CALLS	#3, XPOS\$OPEN	:	1566
		1C	6B	D4	000FC	CLRL	CHROUT	:	1573
			A7	9F	000FE	PUSHAB	TOCOOB+28	:	
			01	DD	00101	PUSHL	#1	:	
00000000G	00	00000000G	8F	DD	00103	PUSHL	#DSRTOCS TMS11	:	
		00000000G	03	FB	00109	CALLS	#3, LIB\$SIGNAL	:	
			EF	9F	00110	PUSHAB	STR\$FAILURE	:	1583
			7E	D4	00116	CLRL	-(SP)	:	
		24	59	DD	00118	PUSHL	R9	:	
			A8	9F	0011A	PUSHAB	\$STR\$STRING	:	
00000000G	EF		7E	D4	0011D	CLRL	-(SP)	:	
	50		05	FB	0011F	CALLS	#5, XST\$COPY	:	
	6B		69	3C	00126	MOVZWL	TMPSTR, R0	:	
44	A7		50	C0	00129	ADDL2	R0, CHROUT	:	
2C	A7		69	9E	0012C	MOVAB	\$IOB\$OUTPUT, IOB\$+68	:	
			07	90	00130	MOVB	#7, IOB\$+44	:	
			5A	DD	00134	PUSHL	R10	:	
			7E	D4	00136	CLRL	-(SP)	:	
00000000G	EF		57	DD	00138	PUSHL	R7	:	
05 00000000G	EF		03	FB	0013A	CALLS	#3, XPOS\$PUT	:	
FEB2	CF		01	E1	00141	BBC	#1, CMDBLK, 5\$	:	
		00000000G	00	FB	00149	CALLS	#0, SPLIT	:	
			EF	9F	0014E	PUSHAB	STR\$FAILURE	:	1584
			7E	D4	00154	CLRL	-(SP)	:	
		2C	59	DD	00156	PUSHL	R9	:	
			A8	9F	00158	PUSHAB	\$STR\$STRING	:	
00000000G	EF		7E	D4	0015B	CLRL	-(SP)	:	
	50		05	FB	0015D	CALLS	#5, XST\$COPY	:	
	6B		69	3C	00164	MOVZWL	TMPSTR, R0	:	
44	A7		50	C0	00167	ADDL2	R0, CHROUT	:	
2C	A7		69	9E	0016A	MOVAB	\$IOB\$OUTPUT, IOB\$+68	:	
			07	90	0016E	MOVB	#7, IOB\$+44	:	
			5A	DD	00172	PUSHL	R10	:	
			7E	D4	00174	CLRL	-(SP)	:	
00000000G	EF		57	DD	00176	PUSHL	R7	:	
05 00000000G	EF		03	FB	00178	CALLS	#3, XPOS\$PUT	:	
FE74	CF		01	E1	0017F	BBC	#1, CMDBLK, 6\$	:	
			00	FB	00187	CALLS	#0, SPLIT	:	



FORMAT  
V04-000

FORMAT - generate formatted output lines  
SPLIT - start new output file for tms if necess

K 15  
16-Sep-1984 00:34:26  
14-Sep-1984 13:06:27

VAX-11 Bliss-32 V4.0-742  
[RUNOFF.SRC]FORMAT.BLI;1

Page 36  
(5)

04 0018C 6\$: RET

; 1587

; Routine Size: 397 bytes, Routine Base: \$CODE\$ + 0985

: 1117 1588 1  
: 1118 1589 1 END  
: 1119 1590 0 ELUDOM

! End of module

.EXTRN LIB\$SIGNAL

# PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	56	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	2834	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$SPLITS	177	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

# Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	202	34	252	00:00.1

# COMMAND QUALIFIERS

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:FORMAT/OBJ=OBJ\$:FORMAT MSRC\$:FORMAT/UPDATE=(ENH\$:FORMAT)

; Size: 2834 code + 233 data bytes  
; Run Time: 01:07.9  
; Elapsed Time: 02:31.0  
; Lines/CPU Min: 1404  
; Lexemes/CPU-Min: 61791  
; Memory Used: 594 pages  
; Compilation Complete



0341

AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY